IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Patent Application of

LINK   Atty. Ref.: 723-1443

Serial No. 10/690,818   TC/A.U.: 3714

Filed: October 23, 2003   Examiner: Nguyen, Dat

For: HAND-HELD VIDEO GAME PLATFORM EMULATION

* *

October 16, 2008

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

AMENDED APPEAL BRIEF IN RESPONSE TO NOTICE OF NON-COMPLIANT APPEAL BRIEF

Sir:

Appellant respectfully submits its Amended Appeal Brief under Rule 41.37 to perfect its appeal from the 1/31/08 Final Rejection. A request for any necessary extension of time under Rule 136(a) is hereby requested up through the date of the filing of this Appeal Brief.

- 1 -

TABLE OF CONTENTS

<u>REAL PARTY IN INTEREST</u>

The real party in interest is Nintendo Co., Ltd., a corporation of the country of Japan.

RELATED APPEALS AND INTERFERENCES

This application is a divisional of USP 6,672,963.

STATUS OF CLAIMS

Claims 1-5, 7-13, and 15-16 have been canceled. Applicant appeals

the 31 January 2008 Final Rejection of claims 6, 14 and 17-58 under 35

USC 134(a). A Notice of Appeal was filed on April 30, 2008.

STATUS OF AMENDMENTS

No amendments have been filed after final rejection.

SUMMARY OF CLAIMED SUBJECT MATTER

The claimed subject matter is directed to an improvement in emulating handheld video game devices of the type that includes a handheld housing having an electronic display thereon and a processor therein that runs a video game software image out of a page-based read only memory (ROM) to present interactive displays on said electronic display of animated video game play in response to user inputs.

In the disclosed exemplary illustrative non-limiting implementation, ROM pages 112 stored in random access memory as shown in Figure 4 emulate a game cartridge read only memory array. Spec. at Para. 83 page 20. In the exemplary illustrative non-limiting implementation, the ROM array is twice as big as the actual ROM pages since the bottom half is always duplicated. Id.; see Figure 5. Description beginning at paragraph 88 (pages 21-22) discloses "duplicating each allocated ROM page to facilitate page selection and reduce page swapping."

6. A method of emulating a handheld video game device of the type that includes a handheld housing having an electronic display thereon **{Fig. 1A; ¶18}**, said handheld housing having therein a processor **{¶56}** that runs a video game software image out of a page-based read only memory

(ROM) **{¶60}** to present interactive displays on said electronic display of animated video game play in response to user inputs **{¶54}**, the method comprising:

executing a video game device emulator program on a target computing device different from said handheld video game device including said handheld housing having said electronic display thereon **{Fig. 2A, ¶54}**, said target computing device being capable of displaying graphical information on a target computing device display **{Fig. 2A, ¶54}**, said target computing device having read/write memory **{Fig. 2 item 68}** and receiving user inputs **{¶57}**, said executing video game device emulator program controlling said target computing device to at least in part emulate said handheld video game device so as to at least in part enable said target computing device to run said video game software and present interactive displays of said animated video game play on said target computing device display in response to user inputs to said target computing platform **{¶54}**;

modeling at least some display timing activities of said handheld video game device electronic display on said target computing device **{¶[0059]}**;

processing, with said emulator program executing on said target computing device, said video game software image capable of being

executed on said handheld video game device processor within said

handheld housing that runs video game software to present interactive

displays on said electronic display of animated video game play in

response to user inputs **{Fig. 2A, block 78; ¶54}**; and

generating a real time interactive video game presentation on said

target computing device display at least in part in response to said

processed video game software image and said modeled display timing

activities **{Fig. 2A; ¶54}**,

wherein said video game software image comprises multiple ROM

pages **{Fig. 5}** and said method further includes said emulator program

providing a pointer table system **{Fig. 15}** that allocates emulated ROM

pages in said target computing device read/write memory and duplicates at

least a portion of said allocated emulated ROM pages across said ROM

pages to facilitate page selection and reduce page swapping **{¶83, 89}**.

14. An emulator that emulates in software, at least a portion of

handheld video game platform hardware of the type including a handheld

housing having an electronic display thereon **{Fig. 1A; ¶18}**, said handheld

housing having therein a processor that runs video game software **{¶56}** ,

said handheld housing accepting a pluggable memory cartridge including a

read only memory (ROM) therein providing a video game software image

comprising multiple ROM pages {¶60}, said handheld video game platform

processor executing said video game software image from said multiple

ROM pages to present interactive displays on said electronic display of

animated video game play in response to user inputs, said emulator

comprising:

a target platform different from said handheld video game platform

hardware, said target platform including a processor that loads and

executes emulation software {Fig. 3, ¶[0064]}, parses and processes a

ROM page based video game software image capable of being executed

on said handheld video game platform hardware processor {Fig. 2A}, and

generates an audio-visual real time interactive presentation in response to

said image {¶54},

wherein said video game software image comprises multiple ROM

pages {Fig. 5} and said emulator provides a pointer table system that

allocates emulated ROM pages in target platform read/write memory and

duplicates at least a portion of said allocated emulated ROM pages across

said allocated ROM pages to facilitate paging operations and reduce page

swapping {Fig. 5} {¶83, 89}.

18.   The method of claim 6 wherein said handheld display comprises

a liquid crystal display {p. 11, lines 1-2} and said modeling comprises

1385328

modeling a virtual liquid crystal display controller state machine **{Fig. 8 }**

corresponding to said handheld liquid crystal display to maintain real time

synchronization with events as they would occur on said handheld video

game device **{¶92}**.

19. The method of claim 6 further including using hardware-assisted

BLIT memory transfer operations to efficiently transfer graphics information

{¶[0018], lines 11-12}.

23. The method of claim 6 further including using said pointer table

system to control memory access by remapping memory access

instructions into function calls **{p. 7, lines 3-5}**.

24. The method of claim 6 further including using said pointer table

system to implement a read only memory protection function to eliminate

overwriting of read only memory **{p. 7, lines 6-7}**.

28. The method of claim 6 further including performing no-operation

look-ahead to avoid wasting processing time in no-operation loops **{p. 7,**

lines 16-17}.

29. The method of claim 6 further including modeling each handheld

video game device native instruction register as a union of byte, word and

long register formats **{p. 7, lines 21-22}**.

37.   A storage device storing emulation software for emulating at

least a portion of the functionality of handheld video game platform

hardware on a portable handheld battery-operated computing device

different from said handheld video game platform hardware **{Fig. 1A; ¶18}**,

said handheld video game platform hardware of the type that includes a

handheld housing having an electronic display thereon**{Fig. 1A; ¶18**, said

handheld housing having therein a processor that runs a video game

software image out of a page-based read only memory (ROM) to present

interactive displays on said electronic display of animated video game play

in response to user inputs**{¶60}**, said portable handheld battery-operated

computing device of the type including a display unit having a

predetermined display area and further including a processor that loads

and executes said stored emulation software to enable video game play on

said portable handheld battery-operated computing device**{¶54}**, said

storage device storing:

first emulation program instructions that process a page-based read

only memory (ROM) video game software image capable of being

executed on said handheld video game platform hardware, said video

game software image comprising multiple ROM pages **{Fig. 5}{¶60}**;

1385328

second emulation program instructions that generate an audio-visual

real time interactive presentation at least in part in response to said video

game software image **{Fig. 2A}**, said second emulation program

instructions displaying an animated character visual part of said audio-

visual presentation on a subset of said display unit predetermined display

area**{¶54}**; and

third emulation program instructions that provide a pointer table

system **{Fig. 15}** which allocates ROM pages in target platform read/write

memory and duplicates at least a portion of said allocated ROM pages

across said pages to facilitate paging operations and reduce page

swapping **{p. 8, lines 3-4}{Fig. 5} {¶83, 89}**.

38. The method of claim 6 further including using said ROM pages to

emulate read only memory arrays within a ROM-based pluggable game

cartridge **{Fig. 4; ¶[0083]}**.

39. The method of claim 6 further including allocating, in random

access memory, at least twice the space occupied by ROM pages in the

handheld video game device, and duplicating half of each page allocated in

random access memory **{¶[0083]}**.

40. The method of claim 6 wherein said handheld video game

device is adapted for use with a pluggable game cartridge ROM having

1385328

ROM banks, and said emulator emulates each of said ROM banks with a different RAM page **{Fig. 5; ¶[0088]}**.

41. The method of claim 6 further including using a ROM selection pointer to select a current ROM page and a ROM page count register to specify the number of emulated ROM pages that have been loaded **{Fig. 5; ¶[0089}**.

42. The method of claim 6 further including using a no-write functional module to protect allocated emulated ROM space so that inadvertent write instructions do not succeed in overwriting emulated read only memory **{Fig. 5; ¶[0089}**,

43. The method of claim 6 further including using a no-write function to protect emulated ROM space from being written to, thus making sure the emulated ROM space is emulated as read only memory rather than read-write memory **{Fig. 5; ¶[0089}**.

44. The method of claim 6 further including using function pointers to implement no-write allocated ROM space protection **{¶[0067]}**.

45. The emulator of claim 14 wherein said emulator uses said allocated ROM pages to emulate the cartridge read only memory arrays **{Figs. 4-5}**.

46. The emulator of claim 14 wherein said emulator allocates, in random access memory, at least twice the space occupied by the ROM pages in the handheld video game memory cartridge, and duplicates half of each page allocated in random access memory **{¶[0083]}**.

47. The emulator of claim 14 wherein said handheld video game pluggable game cartridge ROM has ROM banks, and said emulator emulates each of said ROM banks with a different RAM page **{Fig. 5; ¶[0088]}**.

48. The emulator of claim 14 wherein said emulator uses a ROM selection pointer to select the current ROM page and a ROM page count register to specify the number of ROM pages that have been loaded **{Fig. 5}**.

49. The method of claim 14 wherein said emulator includes a no-write functional module to protect the allocated ROM space so that inadvertent write instructions do not succeed in overwriting emulated read only memory**{Fig. 5; ¶[0089]}**,

50. The emulator of claim 14 further including a no-write function to protect emulated ROM from being written to, thus making sure this memory segment is emulated as read only memory rather than read-write memory **{Fig. 3; ¶[0071]}**.

51. . The emulator of claim 14 further including function pointers that implement no-write register handling functions **{Fig. 3; ¶[0071]}**.

52 . The storage device of claim 37 further including instructions that use said ROM pages to emulate cartridge read only memory arrays **{Figs. 4-5}**.

53. The storage device of claim 37 further including instructions that allocate, in random access memory, at least twice the space occupied by the ROM pages in the handheld video game platform hardware, and duplicating half of each page allocated in random access memory **{¶[0083]}**.

54. The storage device of claim 37 wherein said handheld video game platform hardware is adapted for use with a pluggable game cartridge ROM having ROM banks, and said emulator further includes instructions that emulate each of said ROM banks with a different RAM page **{Fig. 5; ¶[0088]}**.

55. The storage device of claim 37 further including instructions that use a ROM selection pointer to select the current ROM page and a ROM page count register to specify the number of ROM pages that have been loaded **{Fig. 5}**.

56.   The storage device of claim 37 further including no-write functional module instructions to protect the allocated ROM space so that inadvertent write instructions do not succeed in overwriting emulated read only memory **{Fig. 5; ¶[0089]}**.

57.   The storage device of claim 37 further including instructions that use a no-write function to protect emulated ROM from being written to, thus making sure this memory segment is emulated as read only memory rather than read-write memory **{Fig. 3; ¶[0071]}**.

58.   The storage device of claim 37 further including instructions that use function pointers to implement no-write register handling functions **{Fig. 3; ¶[0071]}**.

GROUNDS OF REJECTION TO BE REVIEWED

1. Are claims 6, 14, 18, 23, 27, 29, 32 and 35-58 obvious under 35 USC 103(a) over Snes9x: The Portable Super Nintendo Entertainment System Emulator v1.19 readme.txt dated June 5, 1999 ("Snes9x") in view of Dahl et al. (US 5,949,985) ("Dahl") and further in view of Gameboy98?

2. Is claim 17 obvious under 35 USC 103(a) over Snes9x in view of Dahl and GameBoy98 and Nishiumi et al. (US 6,007,428)?

3. Is claim 19 is not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and Munshi (US 6,084,600)?

4. Is claim 20 is not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and Mackey et al. (US 5,153,577)?

5. Is claim 21 is not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and Sterchi (US 2003/0207712)?

6. Is claim 22 is not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and Morley (US 5,781,758)?

7. Is claim 24 is not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and Woigt, Gunter, "Z80-68K-v150 Z80 Engine written in 68020 assembler for inclusion in C/C++ projects" (12/25/99)?

8. Are claims 25 and 26 are not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and Hannah (US 4,771,279)?

1385328

9. Is claim 28 is not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and in view of Mullarkey et al. (6,192,446)?

10. Is claim 30 is not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and Farber et al. (US 5,903,760)?

11. Is claim 31 is not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and Traut (US 5,790,825)?

12. Is claim 33 is not obvious under 35 USC 103 over Snes9x in view of Dahl, GameBoy98, Munshi (US 6,084,600) and Duruoz et al. (US 6,658,056)?

13. Is claim 34 is not obvious under 35 USC 103 over Snes9x in view of Dahl, GameBoy98 and Reed et al. (6,058,288)?

ARGUMENT

1. Claims 6, 14, 18, 23, 27, 29, 32 and 35-58 are not obvious under 35 USC 103(a) over Snes9x: The Portable Super Nintendo Entertainment System Emulator v1.19 readme.txt dated June 5, 1999 ("Snes9x") in view of Dahl et al. (US 5,949,985) ("Dahl") and further in view of Gameboy98

Independent claim 6 recites in combination "wherein said video game software image comprises multiple ROM pages and said method further includes said emulator program … that allocates emulated ROM pages in said target computing device read/write memory and <u>duplicates at least a portion of said allocated emulated ROM pages across said ROM pages</u> to facilitate page selection and reduce page swapping."   See also independent claims 14 and 37 which each include a similar limitation.

The Examiner states that "Snes9x is silent regarding the ROM pages."   Final Rejection at 3. But it is far from silent. The Super Nintendo Entertainment System that the Snes9x emulates has a ROM cartridge based video game machine including ROM banks. Therefore, the Snes9x emulator reads those ROM files from an optical or magnetic disk and stores them in main memory. See e.g. page 1 comment "Added ROM check sum calculation and testing code – Snes9x can now detect pure, corrupt or hacked ROMs"   The Snes9x documentation also provides details concerning "ROM image format options on pages 8-9 (e.g., interleaved or

alternate interleaved format, Hi-ROM or Lo-ROM memory map, header

options, background priority swapping, H-DMA emulation, etc.). See also

page 11 ("Problems with ROMs") and page 12 ("Converting ROM Images.")

Despite these numerous mentions of storing ROM data, Snes9x provides

no suggestion to duplicate "at least a portion of said allocated emulated

ROM pages across said ROM pages …." as claimed.

The Examiner relies on Dahl et al for the missing teachings.

However, Dahl does not disclose duplicating "at least a portion of said

allocated emulated ROM pages across said ROM pages" as recited in

claim 6.   The Examiner appears to be contending that Dahl's level 1 (L1)

caching of instructions meets this claim limitation. This is incorrect.

Applicant's claims are not directed simply to making a copy per se of

portions of RAM data in cache memory (which is an inherent operation of

almost any personal computer). To the contrary, applicant's claims

specifically require duplicating at least a portion of the allocated emulated

ROM pages ACROSS the ROM pages. Dahl provides no teaching or

suggestion to do this. See KSR v. Teleflex, 550 U.S. ___, 127 S. Ct. 1727

(2007) ("[R]ejections on obviousness grounds cannot be sustained by mere

conclusory statements; instead there must be some articulated reasoning

with some rational underpinning to support the legal conclusion of

obviousness.")

For example, Dahl's Figure 5 cache memory is a separate memory

used just for caching. In contrast, applicant's claim require duplication to

occur "across said pages." Dahl clearly does not teach or suggest the

current claim language with his conventional cache memory teachings.

Dahl was solving a different problem of emulating a hard disk drive file

system and says nothing about how to efficiently emulate a page-based

Read Only Memory of a handheld video game system.

The Examiner states at page 4 of the Office Action that it would have

been obvious to duplicate allocated emulated ROM pages across ROM

pages as claimed because, the Examiner alleges, "applicant has not

disclosed that duplicating of ROM pages across said ROM pages provides

an advantage, is used for a particular purpose, or solves a stated problem."

Office Action at 4-5. The Examiner is mistaken. See page 22 paragraph

89 of applicant's specification which states that such feature facilitates

page selection and reduces page swapping.

Inconsistently, the Examiner urges Dahl's instruction cache to be

equated with applicant's claim feature because they allegedly would

perform equally well "because both duplications perform the same function

of reducing page swapping." Office Action at 5. The Examiner is mistaken here as well. Dahl's L1 instruction caching has as its purpose to reduce the number of instruction fetches by the processor from main memory. See col. 4, lines 4-10; col. 6 lines 45-50 ("only a bilevel memory hierarchy (i.e., L1 cache and memory) and therefore a higher instruction latency when an instruction cache miss occurs.") Dahl's disclosure does not in any way suggest or give any motivation for duplicating certain portions of allocated ROM pages across multiple ROM pages as applicants have claimed.

Furthermore, applying Dahl's teachings to modify Snes9x would merely result in a processor L1 cache containing instructions derived from an SNES ROM file. Applicants assume that the various personal computers discussed in the Snes9x article have L1 caches since such L1 caches are very common in personal computer processor architectures. However, such L1 caches in no way teach or suggest applicant's claimed subject matter concerning ROM page allocation.

The Examiner further contends that Super Nintendo Entertainment System is a "handheld video game system" because a player holds game controllers in their hands. However, applicant's claim limitations are more detailed. The Examiner has not established the requisite correspondence to make a prima facie showing of obviousness. As a backstop, the

Examiner further relies on Gameboy98 as disclosing an emulator for a handheld video game system. However, none of the three applied references teaches or suggests duplicating "at least a portion of said allocated emulated ROM pages across said ROM pages" as recited in claims 6, 14 and 17.

Applicant's dependent claim 23 requires in combination "using said pointer table system to control memory access by remapping memory access instructions into function calls." Claim 6 also requires a pointer table system. This is not taught or suggested by Dahl. The Examiner contends that Dahl teaches this at col. 5 lines 11-43. But that portion of Dahl merely teaches a translated addressing mode – not use of a pointer table system to remap memory access instructions into function calls as recited in claim 23. The point of applicant's claim 23 limitation is not merely to remap memory addresses but rather to remap memory access instructions into function calls. The Examiner also dismisses the claimed "pointer table system" recitation of claim 6 as allegedly being taught in Snes9x or Dahl. However, applicant can find no mention in Snes9x or Dahl of such a pointer table system as claimed for allocating emulated ROM pages.

1385328

With respect to claim 29 ("modeling each handheld video game device native instruction register as a union of byte, word and long register formats"), the Examiner has cited no factual evidence in support of a hindsight conclusion that this would have been obvious. See paragraph 111 of applicants' specification ("The three structures 356, 358, 360 are bundled into a union so that emulator 100 can access a particular register as a byte, a word or a long word as needed.") The Examiner appears to be ignoring or overlooking the word "union."

With respect to the rejection of claim 38, applicant's do not challenge the Examiner's position that Snes9x can read "ROM" files that came from Super Nintendo Entertainment System ROM cartridges. However, this claim in combination provides further distinctions over Dahl's L1 instruction cache teachings. See also dependent claim 40.

The Examiner's attempted application of Dahl to reject dependent claims 39-41, 46-48 and 53-55 is especially strained. Claims 39, 46 and 53 each require allocating in RAM at least twice the space occupied by ROM pages and duplicating half of each page allocated in RAM. The Examiner concedes that Dahl does not disclose the "half of each page allocated in RAM" limitation, but argues that he can ignore this limitation because applicant has alleged failed to disclose that this feature provides an

advantage, is used for a particular purpose or solves a stated problem.

Office Action at 8. This is incorrect. See specification portions cited to

above. The Examiner then speculates that Dahl's instruction duplication

and applicant's claimed subject matter would "perform equally well …"

Even if the Examiner were correct, they are still different and therefore the

argument does not establish the requisite prima facie case of obviousness

under 35 USC 103. Furthermore, the Examiner has lost sight of the fact

that Dahl does not teach duplicating anything across each of multiple ROM

pages allocated in RAM.

Claims 40, 47 and 54 each require in combination emulating each of

plural ROM banks in a pluggable game cartridge ROM with a different RAM

page. There is absolute no teaching or suggestion in Dahl to do this. The

Examiner contends that Dahl teaches the use of RAM pages for each set of

instructions which he equates to a ROM bank, citing col. 5 line 1 to col. 6

line 67. Office Action at 9. However, there is only one System 36

instruction set (which has 138 opcodes). See Figure 4 of Dahl and col. 5

line 66 and following. Dahl's L1 caching thus does not teach "the use of

RAM pages for each set of instructions (ROM bank)" as the Examiner

contends. Furthermore, the Examiner's contention that it would have been

obvious to segregate ROM banks into separate RAM pages "in order to

1385328

break them down into more manageable chucks [sic; chunks] of data"

contradicts the Examiner's other contention that it would have been

obvious to duplicate data across pages in order to reduce page swapping.

If the sole aim was to reduce page swapping, one would want to reduce the

number of RAM pages as much as possible – not segregate data across

more RAM pages than are actually needed to store the data.

Claims 41, 48 and 55 are directed to using a ROM selection pointer

and ROM page count register. The Examiner relies on Dahl for these

teachings as well, but Dahl does not disclose ROM pages let alone ROM

page count registers.

With respect to claims 42-45, 49-52 and 56-58 directed to a no-write

function to protect emulated ROM space from being written to, the

Examiner does not even cite a prior art teaching but simply alleges

obviousness without any support. In fact, Snes9x supplies a suggestion for

allowing writing to ROM space in order to support "cheat options". See

page 8. Therefore, the Examiner's rationale that ROM data cannot be

written to in the original platform and therefore should not be written to

when stored in RAM is not well taken.

2. Claim 17 is not obvious under 35 USC 103(a) over Snes9x in view of Dahl and GameBoy98 and Nishiumi et al. (US 6,007,428)

Claim 17 is patentable for at least the same reasons set forth above

that claim 6 is patentable.

3. Claim 19 is not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and Munshi (US 6,084,600)

Claim 19 is patentable for at least the same reasons set forth above

that claim 6 is patentable. Furthermore, applicant points out that Munshi

criticizes "bitblit" approaches and does not appear to disclose "hardware-

assisted" as claimed.

4. Claim 20 is not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and Mackey et al. (US 5,153,577)

Claim 20 is patentable for at least the same reasons set forth above

that claim 6 is patentable.

5. Claim 21 is not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and Sterchi (US 2003/0207712)

Claim 21 is patentable for at least the same reasons set forth above

that claim 6 is patentable.

6. Claim 22 is not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and Morley (US 5,781,758)

Claim 22 is patentable for at least the same reasons set forth above that claim 6 is patentable.

7. Claim 24 is not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and Woigt, Gunter, "Z80-68K-v150 Z80 Engine written in 68020 assembler for inclusion in C/C++ projects" (12/25/99)

Claim 24 is patentable for at least the same reasons set forth above that claim 6 is patentable.

8. Claims 25 and 26 are not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and Hannah (US 4,771,279)

Claims 25 & 26 are patentable for at least the same reasons set forth above that claim 6 is patentable.

9. Claim 28 is not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and in view of Mullarkey et al. (6,192,446)

Claim 28 is patentable for at least the same reasons set forth above that claim 6 is patentable. Furthermore, the Examiner appears to be overlooking the "no-operation" limitation of claim 28 in equating this claimed feature with Mullarkey's instruction look-ahead circuit. The look-ahead circuit of the applied reference attempts to predict what commands are expected in the near future; it does not avoid "wasting processing time in no-operation loops" as applicant has recited in claim 28.

10. Claim 30 is not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and Farber et al. (US 5,903,760)

Claim 30 is patentable for at least the same reasons set forth above that claim 6 is patentable.

11. Claim 31 is not obvious under 35 USC 103 over Snes9x in view of Dahl and GameBoy98 and Traut (US 5,790,825)

Claim 31 is patentable for at least the same reasons set forth above that claim 6 is patentable.

12. Claim 33 is not obvious under 35 USC 103 over Snes9x in view of Dahl, GameBoy98, Munshi (US 6,084,600) and Duruoz et al. (US 6,658,056)

Claim 33 is patentable for at least the same reasons set forth above that claim 6 is patentable.

13. Claim 34 is not obvious under 35 USC 103 over Snes9x in view of Dahl, GameBoy98 and Reed et al. (6,058,288)

Claim 34 is patentable for at least the same reasons set forth above that claim 6 is patentable.

Respectfully submitted,

NIXON & VANDERHYE P.C.

By: /Robert W. Faris/
                Robert W. Faris
                Reg. No. 31,352

RWF:ejs
901 North Glebe Road, 11th Floor
Arlington, VA 22203-1808
Telephone: (703) 816-4000
Facsimile: (703) 816-4100
rwf@nixonvan.com

CLAIMS APPENDIX

1-5. (Canceled)

6. A method of emulating a handheld video game device of the type that includes a handheld housing having an electronic display thereon, said handheld housing having therein a processor that runs a video game software image out of a page-based read only memory (ROM) to present interactive displays on said electronic display of animated video game play in response to user inputs, the method comprising:

executing a video game device emulator program on a target computing device different from said handheld video game device including said handheld housing having said electronic display thereon, said target computing device being capable of displaying graphical information on a target computing device display, said target computing device having read/write memory and receiving user inputs, said executing video game device emulator program controlling said target computing device to at least in part emulate said handheld video game device so as to at least in part enable said target computing device to run said video game software and present interactive displays of said animated video game play on said target computing device display in response to user inputs to said target computing platform;

modeling at least some display timing activities of said handheld video game device electronic display on said target computing device;

processing, with said emulator program executing on said target computing device, said video game software image capable of being executed on said handheld video game device processor within said handheld housing that runs video game software to present interactive displays on said electronic display of animated video game play in response to user inputs; and

generating a real time interactive video game presentation on said target computing device display at least in part in response to said processed video game software image and said modeled display timing activities,

wherein said video game software image comprises multiple ROM pages and said method further includes said emulator program providing a pointer table system that allocates emulated ROM pages in said target computing device read/write memory and duplicates at least a portion of said allocated emulated ROM pages across said ROM pages to facilitate page selection and reduce page swapping.

7-13. (Canceled)

1385328

14. An emulator that emulates in software, at least a portion of handheld video game platform hardware of the type including a handheld housing having an electronic display thereon, said handheld housing having therein a processor that runs video game software, said handheld housing accepting a pluggable memory cartridge including a read only memory (ROM) therein providing a video game software image comprising multiple ROM pages, said handheld video game platform processor executing said video game software image from said multiple ROM pages to present interactive displays on said electronic display of animated video game play in response to user inputs, said emulator comprising:

a target platform different from said handheld video game platform hardware, said target platform including a processor that loads and executes emulation software, parses and processes a ROM page based video game software image capable of being executed on said handheld video game platform hardware processor, and generates an audio-visual real time interactive presentation in response to said image,

wherein said video game software image comprises multiple ROM pages and said emulator provides a pointer table system that allocates emulated ROM pages in target platform read/write memory and duplicates at least a portion of said allocated emulated ROM pages across said

allocated ROM pages to facilitate paging operations and reduce page

swapping.

15-16. (Canceled)

17. The method of claim 6 wherein said target computing device

display comprises a liquid crystal display.

18. The method of claim 6 wherein said handheld display comprises

a liquid crystal display and said modeling comprises modeling a virtual

liquid crystal display controller state machine corresponding to said

handheld liquid crystal display to maintain real time synchronization with

events as they would occur on said handheld video game device.

19. The method of claim 6 further including using hardware-assisted

BLIT memory transfer operations to efficiently transfer graphics information.

20. The method of claim 6 further including using a pre-computed

translation table that translates native platform graphics character formats.

21. The method of claim 6 further including emulating registers and

hardware-based memory structures within the handheld video game device

in random access memory under software control.

22. The method of claim 6 further including using a jump table to

efficiently parse incoming binary instruction formats.

1385328

23. The method of claim 6 further including using said pointer table system to control memory access by remapping memory access instructions into function calls.

24. The method of claim 6 further including using said pointer table system to implement a read only memory protection function to eliminate overwriting of read only memory.

25. The method of claim 6 wherein said modeling includes using a state machine defining at least a horizontal blank state and a vertical blank state.

26. The method of claim 25 further including providing a cycle timer to determine when a modeled state has expired and transition to a new state is desired.

27. The method of claim 6 further including selectively skipping frames while maintaining execution of instructions to maintain state information while minimizing game play slowdowns.

28. The method of claim 6 further including performing no-operation look-ahead to avoid wasting processing time in no-operation loops.

29. The method of claim 6 further including modeling each handheld video game device native instruction register as a union of byte, word and long register formats.

30. The method of claim 6 further including modeling handheld video game device native instruction CPU flags to allow efficient updating after operations are performed by the target computing device.

31. The method of claim 6 further including mapping the handheld video game device emulated program counter into at least one target computing device microprocessor general purpose register.

32. The method of claim 6 further including providing an adaptable input controller emulator to receive user inputs from a variety of different user input devices.

33. The method of claim 6 further including using screen memory buffers larger than said predetermined display area to increase paging efficiency by eliminating clipping calculations and using hardware Bitblit to transfer a subset of said memory buffer to display video memory.

34. The method of claim 6 wherein said target computing device comprises an airline seat back controller.

35. The method of claim 6 wherein said target computing device comprises a personal digital assistant (PDA).

36. The method of claim 6 wherein said target computing device comprises a handheld portable computing device.

1385328

37. A storage device storing emulation software for emulating at least a portion of the functionality of handheld video game platform hardware on a portable handheld battery-operated computing device different from said handheld video game platform hardware, said handheld video game platform hardware of the type that includes a handheld housing having an electronic display thereon, said handheld housing having therein a processor that runs a video game software image out of a page-based read only memory (ROM) to present interactive displays on said electronic display of animated video game play in response to user inputs, said portable handheld battery-operated computing device of the type including a display unit having a predetermined display area and further including a processor that loads and executes said stored emulation software to enable video game play on said portable handheld battery-operated computing device, said storage device storing:

first emulation program instructions that process a page-based read only memory (ROM) video game software image capable of being executed on said handheld video game platform hardware, said video game software image comprising multiple ROM pages;

second emulation program instructions that generate an audio-visual real time interactive presentation at least in part in response to said video

game software image, said second emulation program instructions

displaying an animated character visual part of said audio-visual

presentation on a subset of said display unit predetermined display area;

and

third emulation program instructions that provide a pointer table

system which allocates ROM pages in target platform read/write memory

and duplicates at least a portion of said allocated ROM pages across said

pages to facilitate paging operations and reduce page swapping.

38. The method of claim 6 further including using said ROM pages to

emulate read only memory arrays within a ROM-based pluggable game

cartridge.

39. The method of claim 6 further including allocating, in random

access memory, at least twice the space occupied by ROM pages in the

handheld video game device, and duplicating half of each page allocated in

random access memory.

40. The method of claim 6 wherein said handheld video game device

is adapted for use with a pluggable game cartridge ROM having ROM

banks, and said emulator emulates each of said ROM banks with a

different RAM page.

41. The method of claim 6 further including using a ROM selection pointer to select a current ROM page and a ROM page count register to specify the number of emulated ROM pages that have been loaded.

42. The method of claim 6 further including using a no-write functional module to protect allocated emulated ROM space so that inadvertent write instructions do not succeed in overwriting emulated read only memory,

43. The method of claim 6 further including using a no-write function to protect emulated ROM space from being written to, thus making sure the emulated ROM space is emulated as read only memory rather than read-write memory.

44. The method of claim 6 further including using function pointers to implement no-write allocated ROM space protection.

45. The emulator of claim 14 wherein said emulator uses said allocated ROM pages to emulate the cartridge read only memory arrays.

46. The emulator of claim 14 wherein said emulator allocates, in random access memory, at least twice the space occupied by the ROM pages in the handheld video game memory cartridge, and duplicates half of each page allocated in random access memory.

1385328

47. The emulator of claim 14 wherein said handheld video game pluggable game cartridge ROM has ROM banks, and said emulator emulates each of said ROM banks with a different RAM page.

48. The emulator of claim 14 wherein said emulator uses a ROM selection pointer to select the current ROM page and a ROM page count register to specify the number of ROM pages that have been loaded.

49. The method of claim 14 wherein said emulator includes a no-write functional module to protect the allocated ROM space so that inadvertent write instructions do not succeed in overwriting emulated read only memory,

50. The emulator of claim 14 further including a no-write function to protect emulated ROM from being written to, thus making sure this memory segment is emulated as read only memory rather than read-write memory.

51. . The emulator of claim 14 further including function pointers that implement no-write register handling functions.

52 . The storage device of claim 37 further including instructions that use said ROM pages to emulate cartridge read only memory arrays.

53. The storage device of claim 37 further including instructions that allocate, in random access memory, at least twice the space occupied by

the ROM pages in the handheld video game platform hardware, and duplicating half of each page allocated in random access memory.

54. The storage device of claim 37 wherein said handheld video game platform hardware is adapted for use with a pluggable game cartridge ROM having ROM banks, and said emulator further includes instructions that emulate each of said ROM banks with a different RAM page.

55. The storage device of claim 37 further including instructions that use a ROM selection pointer to select the current ROM page and a ROM page count register to specify the number of ROM pages that have been loaded.

56. The storage device of claim 37 further including no-write functional module instructions to protect the allocated ROM space so that inadvertent write instructions do not succeed in overwriting emulated read only memory,

57. The storage device of claim 37 further including instructions that use a no-write function to protect emulated ROM from being written to, thus making sure this memory segment is emulated as read only memory rather than read-write memory.

58.   The storage device of claim 37 further including instructions that use function pointers to implement no-write register handling functions.

1385328

EVIDENCE APPENDIX

None.

RELATED PROCEEDINGS APPENDIX

None.